

XML Signature (XMLSIG) Implementation Specification for NSIF/NITF

*Proposed for Incorporation into
XML Schema/Specification Register
of the
NATO Secondary Imagery Format (NSIF)
and the
National Imagery Transmission Format Standard (NITFS)*

*Version 1.0, 21 November 2009
Administratively Updated 13 September 2013*

(This page intentionally left blank.)

Table of Contents

| | | |
|------------|--------------------------------------|-----------|
| 1.0 | SCOPE..... | 5 |
| 1.1 | Introduction | 5 |
| 1.2 | Purpose..... | 5 |
| 1.3 | Security Considerations | 5 |
| 2.0 | REFERENCES..... | 6 |
| 3.0 | OVERVIEW | 6 |
| 4.0 | XML Signature Definition..... | 7 |
| 4.1 | Data Model..... | 7 |
| 4.2 | Schema | 8 |
| 4.3 | Uniform Resource Identifier..... | 9 |
| 4.4 | Digest Scope | 9 |
| 4.5 | Example | 9 |
| 5.0 | TEST CRITERIA | 11 |
| 5.1 | XML Signature Pack Criteria | 11 |
| 5.2 | XML Signature Unpack Criteria | 11 |
| | Appendix A – XML Schema | 13 |

(This page intentionally left blank.)

1.0 SCOPE

1.1 Introduction

The XML Signature is used to validate the integrity of a NSIF/NITF file and to associate the file with its source. The XML Signature is included in a XML_DATA_CONTENT Data Extension Segment (DES) of an NSIF/NITF file (Reference [4]).

The XML Signature is based on a public key infrastructure that provides digital certificates to uniquely identify a user¹, support non-repudiation, and support entity-specific encryption of information. The infrastructure includes the directory services that issue, store, manage and revoke the certificates.

1.2 Purpose

The XML Signature supports three related objectives: (1) assuring the integrity of the data, (2) reliably making decisions based on information in the NSIF/NITF headers, and (3) non-repudiable identifying the source of an image. Applications may use the XML Signature to confirm that the data in the NSIF/NITF file has not been accidentally or intentionally modified since the signature was attached. Applications may also use XML Signature to confirm the validity of the header information, such as classification and releasing instructions, and confirm its relationship to the image data before making decisions about disseminating an NSIF/NITF file across a security domain. The XML Signature establishes the relationship to the image source through the signature, which supports the authentication of imagery, as well as the assured identification of images from a specific source.

1.3 Security Considerations

The XML Signature establishes integrity mechanisms and reliable binding within an NSIF/NITF file, the trust-level for these mechanisms is no greater than the environment where the labels are applied.

The XML Signature relies on public/private keys to identify an image signer, and is only as reliable as the protection of the private keys, and the process used to match a signature to an entity.

The XML Signature applies to the NSIF/NITF segments, but does not apply to the file header. All decisions requiring validated data, e.g. security release, should be based on the segment headers.

¹ A “user” in this context is a person or an automated mechanism.

2.0 REFERENCES

- [1] MIL-STD-2500C National Imagery Transmission Format Version 2.1 for the National Imagery Transmission Format Standard, Release 1.2, 19 September 2005
- [2] STANAG 4545 ed 2 NATO Secondary Imagery Format (NSIF), Edition 2, 6 May 2013.
- [3] AEDP-4 NATO Secondary Imagery Format (NSIF) STANAG 4545 Implementation Guide, Edition 2, July 2012
- [4] STDI-0002-2_1.0 The Compendium of Data Extension Segments (DES) for the National Imagery Transmission Format (NITF), (APPENDIX F, XML_DATA_CONTENT DES, VERSION 1.0, 3 December 2012)
- [5] IETF RFC 2807 (Extensible Markup Language) XML Signature Requirements, July 2000, <http://www.ietf.org/rfc/rfc2807.txt> (Internet)
- [6] IETF RFC 3275 (Extensible Markup Language) XML-Signature Syntax and Processing, March 2002, <http://www.ietf.org/rfc/rfc3275.txt> (Internet)
- [7] XML Signature Syntax and Processing (Second Edition) W3C Recommendation, 10 June 2008, <http://www.w3.org/TR/xmldsig-core/> (Internet)
- [8] Canonical XML W3C Recommendation, Version 1.0, 15 March 2001, <http://www.w3.org/TR/xml-c14n> (Internet)
- [9] IETF RFC 2437 PKCS #1: RSA Cryptography Specifications, Version 2.0, October 1998, <http://www.ietf.org/rfc/rfc2437.txt> (Internet)
- [10] FIPS PUB 180-3 Secure Hash Standard, Federal Information Processing Standards Publication, Information Technology Laboratory National Institute of Standards and Technology, http://csrc.nist.gov/publications/fips/fips180-3/fips180-3_final.pdf (Internet)

3.0 OVERVIEW

XML signatures apply to a part or parts of a document. For NSIF/NITF files, the parts are the individual image, graphic, text, data and reserved extension segments. XML signatures reference document parts with URIs (Universal Resource Indicators) and fragment identifiers. The NSIF/NITF segments are referenced as shown in Section 6.1

of [1], where “IM001” refers to the first image segment, “TE004” refers to the fourth text segment, etc. The segment sequence number is based on the sequence numbers implicit in the length parameters of the NSIF/NITF file header (e.g. LIn, LTn) as defined in References [1] and [2].

Because the URI references are implicit, rather than explicit, the XML Signature does not support the layering of information. Applications re-writing an NSIF/NITF file must re-order or re-compute digest values based on the order of segments of the re-written files, and re-sign the file.

An NSIF/NITF file may contain one or more XML_DATA_CONTENT DES's, and each DES may contain zero, one or more XML signatures. Each XML signature will include one or more digest values for NSIF/NITF segments. Not all segments in the NSIF/NITF file are necessarily included in the XML signature.

The XML Signature is described in IETF RFC 2807 (Reference [5]) and the W3C XMLDSIG Recommendation (Reference [7]), and specified in IETF RFC 3275 (Reference [6]).

4.0 XML Signature Definition

4.1 Data Model

The XML Signature shall conform to the data model depicted in Figure 1, which is unmodified from the W3C recommendation (Reference [7]).

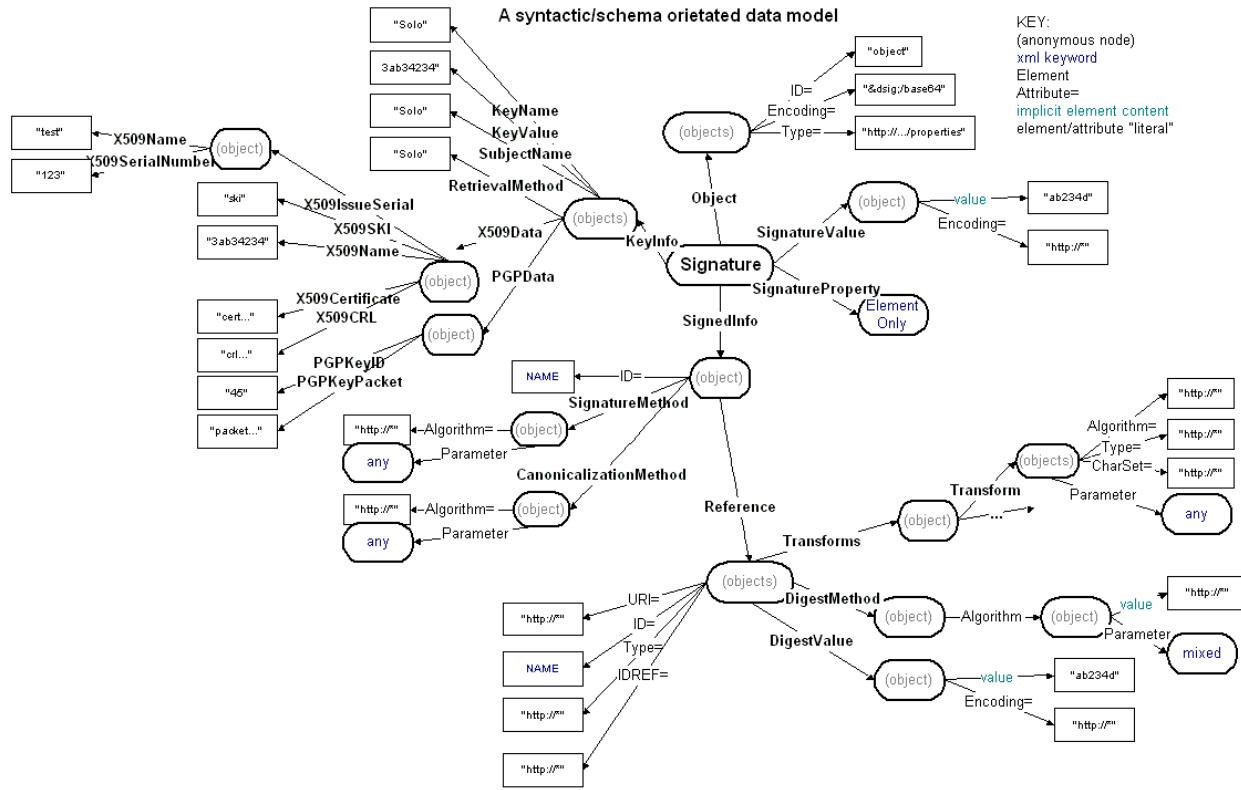


Figure 1 – RDF Data Model

4.2 Schema

The XML Signature shall comply with the XML schema in Appendix A.

The XML Signature shall use the following selections:

- CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"
 - The canonical (physical) representation of the XML Signature is based on Canonical XML Version 1.0 (Reference [8]).
- SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"
 - The XML Signature includes a PKCS1 signature per IETF RFC 2437 (Reference [9]).
- DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
 - The digest values are hashed with the SHA-1 algorithm.

4.3 Uniform Resource Identifier

The XML Signature shall reference segments within the NSIF/NITF file as URIs as shown in [1], e.g. URI="NITF://IM001", where "IM" is the segment type (IM=image segment, SY=symbol (graphic) segment, TE=text segment, DE=data extension segment), and "001" is the 3-digit sequence number implied from the order of the segments in the file (the first image segment is 001, the second is 002, etc.).

4.4 Digest Scope

Each XML Signature digest value shall encompass the subheader and the data field for a single file segment.

4.5 Example

Figure 2 is an example of an XML Signature for an NSIF/NITF file with a single image segment and a single text segment. In this formatted representation of the XML:

- The <SignedInfo> sequence includes the <CanonicalizationMethod Algorithm> and the <SignatureMethod Algorithm>, with a <Reference> for each segment.
- Each <Reference> includes the <DigestMethod Algorithm> and the <DigestValue> (shown as a placeholder in this example).
- The <SignedInfo> is followed by the <SignatureValue> (also shown as a placeholder in this example).
- The <SignatureValue> is followed by the <KeyInfo> which includes the <RSAKeyValue> and the <X509Data> (both are shown unexpanded in this example).

```
<?xml version="1.0" encoding="UTF-8" ?>
- <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
  - <SignedInfo>
    <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
    <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
  - <Reference URI="NITF://IM001">
    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
    <DigestValue>160-bit digest value goes here</DigestValue>
  </Reference>
  - <Reference URI=" NITF://TE001">
    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
    <DigestValue>160-bit digest value goes here</DigestValue>
  </Reference>
</SignedInfo>
<SignatureValue>RSA signature value goes here</SignatureValue>
- <KeyInfo>
  - <KeyValue>
    + <RSAKeyValue>
    </KeyValue>
  + <X509Data>
    </KeyInfo>
</Signature>
```

Figure 2 – XML Signature Example

5.0 TEST CRITERIA

5.1 XML Signature Pack Criteria

The following test criteria apply to the packing (generation) of NSIF/NITF files when using the XML Signature extension, as specified for the XML_DATA_CONTENT DES (Reference [4]).

- All character values in XML_DATA_CONTENT DES shall be in the printable Basic Character Set (BCS)-A character set (space (0x20) through tilde (0x7E)) with eight bits (one byte per character). Note: because of inconsistencies between standards, the use of ECS-A characters is restricted to its BCS-A Subset.
- All data in fields designated as alphanumeric (BCS-A) shall be left justified and padded with spaces as necessary to fill the field.
- All data in numeric (BCS-N) fields shall be right justified and padded with leading zeros as necessary to fill the field.
- All required fields shall be present and contain valid data as defined in the XML_DATA_CONTENT DES specification.
- Conditional fields shall only be present when the conditional parameters of predicate fields so signify. When present, conditional fields shall contain valid data.
- The DESSHTN (Target Namespace) field in the XML_DATA_CONTENT DES subheader shall be populated with the default value for XML Signature (the presence of other XML content may separately require the population of these fields). Other fields may be populated with default or unknown-designator values when the actual value is unavailable.
- The XML Signature is not geographic, and the location fields are not required ((the presence of other XML content may separately require the population of these fields). The DESSHLPG through DESSHABS fields may be omitted from the header.

5.2 XML Signature Unpack Criteria

The following test criteria apply to the unpacking and interpretation of the information provided in the XML Signature with in the XML_DATA_CONTENT DES, including the criteria specified in Reference [4].

- Upon receipt of an NITF/NSIF file that contains one or more DES, an otherwise NSIF/NITF compliant system that is not designed to interpret that DES shall ignore it and properly interpret the other NITF/NSIF file components.

- NSIF/NITF implementations that support the XML_DATA_CONTENT DES shall comply with the minimum conformance requirements identified in the specification of the XML data content (see fields DESSHSDI, DESSHSDV, and DESSHSD)..
- The XML Signature shall be unpacked, expanded (canonicalized), and interpreted from the XML_DATA_CONTENT DES as described in this document.

Appendix A – XML Schema

The XML Signature will comply with the XML Schema listed below. This is unmodified from the W3C recommendation (Reference [7]).

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE schema
  PUBLIC "-//W3C//DTD XMLSchema 200102//EN"
  "http://www.w3.org/2001/XMLSchema.dtd"
[
  <!ATTLIST schema
    xmlns:ds CDATA #FIXED "http://www.w3.org/2000/09/xmldsig#"
  <!ENTITY dsig 'http://www.w3.org/2000/09/xmldsig#'>
  <!ENTITY % p ''>
  <!ENTITY % s ''>
]>

<!-- Schema for XML Signatures
  http://www.w3.org/2000/09/xmldsig#
  $Revision: 1.1 $ on $Date: 2002/02/08 20:32:26 $ by $Author: reagle
$

  Copyright 2001 The Internet Society and W3C (Massachusetts
Institute
  of Technology, Institut National de Recherche en Informatique et en
  Automatique, Keio University). All Rights Reserved.
  http://www.w3.org/Consortium/Legal/

  This document is governed by the W3C Software License [1] as
described
  in the FAQ [2].
[1] http://www.w3.org/Consortium/Legal/copyright-software-19980720
[2] http://www.w3.org/Consortium/Legal/IPR-FAQ-20000620.html#DTD
--&gt;

&lt;schema xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  targetNamespace="http://www.w3.org/2000/09/xmldsig#"
  version="0.1" elementFormDefault="qualified"&gt;

  <!-- Basic Types Defined for Signatures --&gt;

&lt;simpleType name="CryptoBinary"&gt;
  &lt;restriction base="base64Binary"&gt;
  &lt;/restriction&gt;
&lt;/simpleType&gt;

  <!-- Start Signature --&gt;

&lt;element name="Signature" type="ds:SignatureType"/&gt;
&lt;complexType name="SignatureType"&gt;
  &lt;sequence&gt;
    &lt;element ref="ds:SignedInfo"/&gt;
    &lt;element ref="ds:SignatureValue"/&gt;
    &lt;element ref="ds:KeyInfo" minOccurs="0"/&gt;
  &lt;/sequence&gt;
&lt;/complexType&gt;</pre>
```

```
<element ref="ds:Object" minOccurs="0" maxOccurs="unbounded"/>
</sequence>
<attribute name="Id" type="ID" use="optional"/>
</complexType>

<element name="SignatureValue" type="ds:SignatureValueType"/>
<complexType name="SignatureValueType">
<simpleContent>
<extension base="base64Binary">
<attribute name="Id" type="ID" use="optional"/>
</extension>
</simpleContent>
</complexType>

<!-- Start SignedInfo -->

<element name="SignedInfo" type="ds:SignedInfoType"/>
<complexType name="SignedInfoType">
<sequence>
<element ref="ds:CanonicalizationMethod"/>
<element ref="ds:SignatureMethod"/>
<element ref="ds:Reference" maxOccurs="unbounded"/>
</sequence>
<attribute name="Id" type="ID" use="optional"/>
</complexType>

<element name="CanonicalizationMethod"
type="ds:CanonicalizationMethodType"/>
<complexType name="CanonicalizationMethodType" mixed="true">
<sequence>
<any namespace="#any" minOccurs="0" maxOccurs="unbounded"/>
<!-- (0,unbounded) elements from (1,1) namespace -->
</sequence>
<attribute name="Algorithm" type="anyURI" use="required"/>
</complexType>

<element name="SignatureMethod" type="ds:SignatureMethodType"/>
<complexType name="SignatureMethodType" mixed="true">
<sequence>
<element name="HMACOutputLength" minOccurs="0"
type="ds:HMACOutputLengthType"/>
<any namespace="#other" minOccurs="0" maxOccurs="unbounded"/>
<!-- (0,unbounded) elements from (1,1) external namespace -->
</sequence>
<attribute name="Algorithm" type="anyURI" use="required"/>
</complexType>

<!-- Start Reference -->

<element name="Reference" type="ds:ReferenceType"/>
<complexType name="ReferenceType">
<sequence>
<element ref="ds:Transforms" minOccurs="0"/>
<element ref="ds:DigestMethod"/>
<element ref="ds:DigestValue"/>
</sequence>
<attribute name="Id" type="ID" use="optional"/>
```

```
<attribute name="URI" type="anyURI" use="optional"/>
<attribute name="Type" type="anyURI" use="optional"/>
</complexType>

<element name="Transforms" type="ds:TransformsType"/>
<complexType name="TransformsType">
    <sequence>
        <element ref="ds:Transform" maxOccurs="unbounded"/>
    </sequence>
</complexType>

<element name="Transform" type="ds:TransformType"/>
<complexType name="TransformType" mixed="true">
    <choice minOccurs="0" maxOccurs="unbounded">
        <any namespace="#other" processContents="lax"/>
        <!-- (1,1) elements from (0, unbounded) namespaces -->
        <element name="XPath" type="string"/>
    </choice>
    <attribute name="Algorithm" type="anyURI" use="required"/>
</complexType>

<!-- End Reference -->

<element name="DigestMethod" type="ds:DigestMethodType"/>
<complexType name="DigestMethodType" mixed="true">
    <sequence>
        <any namespace="#other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </sequence>
    <attribute name="Algorithm" type="anyURI" use="required"/>
</complexType>

<element name="DigestValue" type="ds:DigestValueType"/>
<simpleType name="DigestValueType">
    <restriction base="base64Binary"/>
</simpleType>

<!-- End SignedInfo -->

<!-- Start KeyInfo -->

<element name="KeyInfo" type="ds:KeyInfoType"/>
<complexType name="KeyInfoType" mixed="true">
    <choice maxOccurs="unbounded">
        <element ref="ds:KeyName"/>
        <element ref="ds:KeyValue"/>
        <element ref="ds:RetrievalMethod"/>
        <element ref="ds:X509Data"/>
        <element ref="ds:PGPData"/>
        <element ref="ds:SPKIData"/>
        <element ref="ds:MgmtData"/>
        <any processContents="lax" namespace="#other"/>
        <!-- (1,1) elements from (0, unbounded) namespaces -->
    </choice>
    <attribute name="Id" type="ID" use="optional"/>
</complexType>
```

```
<element name="KeyName" type="string"/>
<element name="MgmtData" type="string"/>

<element name="KeyValue" type="ds:KeyValueType"/>
<complexType name="KeyValueType" mixed="true">
  <choice>
    <element ref="ds:DSAKeyValue"/>
    <element ref="ds:RSAKeyValue"/>
    <any namespace="#other" processContents="lax"/>
  </choice>
</complexType>

<element name="RetrievalMethod" type="ds:RetrievalMethodType"/>
<complexType name="RetrievalMethodType">
  <sequence>
    <element ref="ds:Transforms" minOccurs="0"/>
  </sequence>
  <attribute name="URI" type="anyURI"/>
  <attribute name="Type" type="anyURI" use="optional"/>
</complexType>

<!-- Start X509Data -->

<element name="X509Data" type="ds:X509DataType"/>
<complexType name="X509DataType">
  <sequence maxOccurs="unbounded">
    <choice>
      <element name="X509IssuerSerial" type="ds:X509IssuerSerialType"/>
      <element name="X509SKI" type="base64Binary"/>
      <element name="X509SubjectName" type="string"/>
      <element name="X509Certificate" type="base64Binary"/>
      <element name="X509CRL" type="base64Binary"/>
      <any namespace="#other" processContents="lax"/>
    </choice>
  </sequence>
</complexType>

<complexType name="X509IssuerSerialType">
  <sequence>
    <element name="X509IssuerName" type="string"/>
    <element name="X509SerialNumber" type="integer"/>
  </sequence>
</complexType>

<!-- End X509Data -->

<!-- Begin PGPData -->

<element name="PGPData" type="ds:PGPDataType"/>
<complexType name="PGPDataType">
  <choice>
    <sequence>
      <element name="PGPKeyID" type="base64Binary"/>
      <element name="PGPKeyPacket" type="base64Binary" minOccurs="0"/>
      <any namespace="#other" processContents="lax" minOccurs="0"
           maxOccurs="unbounded"/>
    </sequence>
  </choice>
</complexType>
```

```
<sequence>
  <element name="PGPKeyPacket" type="base64Binary"/>
  <any namespace="#other" processContents="lax" minOccurs="0"
    maxOccurs="unbounded"/>
</sequence>
</choice>
</complexType>

<!-- End PGPData -->

<!-- Begin SPKIData -->

<element name="SPKIData" type="ds:SPKIDataType"/>
<complexType name="SPKIDataType">
  <sequence maxOccurs="unbounded">
    <element name="SPKISexp" type="base64Binary"/>
    <any namespace="#other" processContents="lax" minOccurs="0"/>
  </sequence>
</complexType>

<!-- End SPKIData -->

<!-- End KeyInfo -->

<!-- Start Object (Manifest, SignatureProperty) -->

<element name="Object" type="ds:ObjectType"/>
<complexType name="ObjectType" mixed="true">
  <sequence minOccurs="0" maxOccurs="unbounded">
    <any namespace="#any" processContents="lax"/>
  </sequence>
  <attribute name="Id" type="ID" use="optional"/>
  <attribute name="MimeType" type="string" use="optional"/> <!-- add a
grep facet -->
  <attribute name="Encoding" type="anyURI" use="optional"/>
</complexType>

<element name="Manifest" type="ds:ManifestType"/>
<complexType name="ManifestType">
  <sequence>
    <element ref="ds:Reference" maxOccurs="unbounded"/>
  </sequence>
  <attribute name="Id" type="ID" use="optional"/>
</complexType>

<element name="SignatureProperties" type="ds:SignaturePropertiesType"/>
<complexType name="SignaturePropertiesType">
  <sequence>
    <element ref="ds:SignatureProperty" maxOccurs="unbounded"/>
  </sequence>
  <attribute name="Id" type="ID" use="optional"/>
</complexType>

<element name="SignatureProperty" type="ds:SignaturePropertyType"/>
<complexType name="SignaturePropertyType" mixed="true">
  <choice maxOccurs="unbounded">
    <any namespace="#other" processContents="lax"/>
  </choice>
</complexType>
```

```
<!-- (1,1) elements from (1,unbounded) namespaces -->
</choice>
<attribute name="Target" type="anyURI" use="required"/>
<attribute name="Id" type="ID" use="optional"/>
</complexType>

<!-- End Object (Manifest, SignatureProperty) -->

<!-- Start Algorithm Parameters -->

<simpleType name="HMACOutputLengthType">
  <restriction base="integer"/>
</simpleType>

<!-- Start KeyValue Element-types -->

<element name="DSAKeyValue" type="ds:DSAKeyValueType"/>
<complexType name="DSAKeyValueType">
  <sequence>
    <sequence minOccurs="0">
      <element name="P" type="ds:CryptoBinary"/>
      <element name="Q" type="ds:CryptoBinary"/>
    </sequence>
    <element name="G" type="ds:CryptoBinary" minOccurs="0"/>
    <element name="Y" type="ds:CryptoBinary"/>
    <element name="J" type="ds:CryptoBinary" minOccurs="0"/>
    <sequence minOccurs="0">
      <element name="Seed" type="ds:CryptoBinary"/>
      <element name="PgenCounter" type="ds:CryptoBinary"/>
    </sequence>
  </sequence>
</complexType>

<element name="RSAKeyValue" type="ds:RSAKeyValueType"/>
<complexType name="RSAKeyValueType">
  <sequence>
    <element name="Modulus" type="ds:CryptoBinary"/>
    <element name="Exponent" type="ds:CryptoBinary"/>
  </sequence>
</complexType>

<!-- End KeyValue Element-types -->

<!-- End Signature -->

</schema>
```